Research on Android Intent Security Detection Based on Machine Learning

Lv Zhuo, Guo Zhimin, Chen cen
State Grid Henan Electric Power Research Institute
Zhengzhou, China
zhuanzhuan2325@sina.com

Abstract—With the rapid development of mobile Internet, android system becomes the absolute overlord of the operating system, which has an unparalleled influence to the smart phone industry, but also closely related with people's lives. However, the feature of open source and lacking of regulatory of Android system led to a lot of security risks. Because of the widespread use of Android Intent Mechanism, the number of vulnerabilities of denial of service and authority to bypass due to the Intent mechanism is far ahead in recent years. In this paper, we focused on the detection of security vulnerabilities based on Intent mechanism and proposed a security detection model based on machine learning, which combines static detection and dynamic detection, and uses machine learning detectors to classify detection the Intent component. This model could be a better implementation of security vulnerability detection to help developers and Android project team to carry out security testing and maintenance better.

Keywords—Android security; Intent Fuzz; Machine learning; Vulnerability mining

I. INTRODUCTION

With the continuous development of the Internet, as well as the promotion of the mobile communication technology (such as 3G, 4G), the mobile Internet has been in a high-speed growth trend. According to the "38th China Internet Development Statistics Report" data released by the China Internet Network Information Center, as of June 2016, the scale of China's Internet users has reached 710 million, within half a year newly added 21.32 million Internet users in total. The Internet penetration rate is 51.7%, up 1.3 percentage points from the end of 2015. Among them, the scale of China's mobile Internet users has reached to 656 million, an increase of more than 36 million 560 thousand people by the end of 2015. The proportion of Internet users who use mobile phone to get online increased from 90.1% to 92.5% from the end of 2015 [1-3]. Specific data are shown in Fig.1 and Fig.2.

However, with the popularity of smart mobile terminals, their various security issues have become more prominent. While people enjoy the convenience of the mobile network, we must also face the current security issues that cannot be ignored. Ali mobile security third quarter report of 2015 shows that Android system vulnerabilities continue to increase in recent years. There are 11630 vulnerabilities in Top 10 applications of 16 industries in all and each application has 73 vulnerabilities on average, which is an increase of 37% compared to the second quarter. The web view remote code execution vulnerability accounted for a large proportion of these vulnerabilities [4-5].

Therefore, the security flaws of system and application will lead to a great deal of information security risks of the user. The security testing will have an irreplaceable value and necessity for the Android system and its application security. The Intent mechanism is a very important technology in Android application development so its in-depth study is very meaningful. [6]

At present, android application security detection technology is mainly divided into two aspects: static detection technology and dynamic detection technology [7-8].

Static detection means in the case of without executing the application, analyzing the application or its source code through the unique analysis tools, etc. The static detection method has a fast detection speed but it could only detect known feature codes and need to constantly update the existing signature database [9].

Dynamic detection means that when the applications run in certain circumstances recording and testing the program behavior and the system state changes to detect the threat of the application [10].

The current mainstream technology for security detection of Android Intent mechanism is the fuzzy test in dynamic analysis, and then supplements the corresponding static...
analysis. Based on these the corresponding security detection framework or system was developed

II. ANDROID INTENT MECHANISM

Intent is a run-time binding mechanism, which can connect two different components in the process of running program. Through the Intent, the program can express a request or wish to Android. Android will select the appropriate components to complete the request according to the content of wish. For example, if an Activity wants to open a Web browser to view the content of a Web page, the Activity only needs to send WEB_SEARCH_ACTION to Android. Android will check the Intent Filter which was declared when the component registered and find Activity of the web browser to browse the web according to the contents of the Intent request. Android’s three basic components - Activity, Service and Broadcast Receiver are activated through the Intent mechanism.

A. The composition of Intent

To transfer data between different Activities, it is necessary to include the corresponding content in the Intent. In general the most basic data should include the parameters shown in TABLE I.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Action</td>
<td>Used to indicate what action to implement, such as ACTION_VIEW, ACTION_EDIT and so on</td>
</tr>
<tr>
<td>Data</td>
<td>Specific data, is generally represented by a Uri variable</td>
</tr>
<tr>
<td>Category</td>
<td>A string contains information about the kind of component handling the Intent. An Intent object can have any category. The Intent class defines many Category constants</td>
</tr>
<tr>
<td>Component</td>
<td>Appoints the class name of the target component of Intent. In general Android will check according to the information of other attributes contained in the Intent, such as action, data / type, category, and ultimately find a matching target components. However, if the component attribute is specified, it will use the specified component directly instead of performing the process of the above-mentioned lookup.</td>
</tr>
<tr>
<td>Extras</td>
<td>Additional information, such as the intent of ACTION_TIMEZONE_CHANGED, has a “time-zone” additional information to indicate the new time zone. Intent objects have a set of put ... () and set ... () methods to set and get additional information. These methods are very similar to Bundle objects. In fact, additional information can use putExtras () and getExtras () as Bundle to read and write.</td>
</tr>
</tbody>
</table>

B. The security detection for the Intent mechanism

At present, there are two kinds of security vulnerabilities caused by Intent. One is authority bypass vulnerability, and the other is the denial of service vulnerability. The authority bypass vulnerability can allow applications or malware get the corresponding system and service privilege to gain some important or sensitive information, such as controlling the user's mobile phone to send spam messages, stealing user privacy data, and even stealing user’s money. The denial of service vulnerability not only can lead to the protection function be bypassed or disabled (such as antivirus applications, security guards, security lock screen, etc.), but also can be used to launch attacks which can cause the application to crash. [11]

Malicious applications or criminals could use the Intent mechanism vulnerabilities in system native applications or installed applications to launch attacks. Such as sending the false intentions to the message application to make the system accept it as the right one, in order to achieve the purpose of deceiving the user, or sending the corresponding Intent to a financial application to transfer the user’s funds.

For the problem of the Intent mechanism, it is possible to detect whether there is security problem by dynamic fuzzy test or static analysis. Static analysis is responsible for the detection of the application authority and the registration information of each component, and its main function is the decompiled source code review. Dynamic fuzzy test randomly generates various types of different Intent according to the collected component information and sends them to the corresponding components to test whether there are loopholes. If the dynamic fuzzy test and static analysis are combined, you will obtain better detection results.

III. RESEARCH ON MACHINE LEARNING ALGORITHMS

A. Classification algorithm

1) Decision tree algorithm

Decision tree is a tree structure (either a binary tree or a non-binary tree). Each non-leaf node represents a test on the feature attribute; each branch represents the output of the feature attribute on a range; and each leaf node stores a category. The process of using decision tree to decide is starts with the root node, then test the corresponding attributes of the items to be classified, and select the output branches according to their values until they reach the leaf nodes. The category that is stored on the leaf node is used as the decision result. As shown in Fig.3.

The construction process of the decision tree does not depend on domain knowledge, and it uses attribute selection metrics to divide the attributes into different classes. The construction of the so-called decision tree is using attribute selection measures to determine the topological structure of each feature attribute. The key step of constructing a decision tree is the split attribute. The so-called split attribute is to divide and construct different branches according to the difference of a feature attribute at a node.

The key content of constructing a decision tree is to make attribute selection measures. The attribute selection metric is a kind of selective split criterion, which is the heuristic method of “best” dividing the data of the training set that the given class marked into the individual class. It determines the choice of topological structure and splitting point. There are many attribute selection measurement algorithms, and in general we
use the top-down recursive divide-and-conquer method. Two commonly used algorithms are ID3 and C4.5.

2) ID3 algorithm:

According to the knowledge of information theory, we know that the smaller the expected information, the greater the information gain, and thus the higher the purity. Therefore, the core idea of ID3 algorithm is to select the attribute which information gain is largest to split. We define a few concepts to be used first.

Let D be the division of the training tuple by category, then the entropy of D is expressed as:

$$\text{info}(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$  \hspace{1cm} (1)

Among them, pi represents the probability of the i-th category appearing in the entire training tuple, and we can estimate by dividing the number of elements belonging to this category by the total number of elements of the training tuple. The actual meaning of entropy is the average amount of information required for class labels of the tuple in D.

Assuming that the training tuple D is divided by the attribute A, then the expected information of dividing D by A is:

$$\text{info}_A(D) = \sum_{|D_j|} \frac{|D_j|}{|D|} \text{info}(D_j)$$  \hspace{1cm} (2)

And the information gain is the difference between the two:

$$\text{gain}(A) = \text{info}(D) - \text{info}_A(D)$$  \hspace{1cm} (3)

ID3 algorithm is to calculate the gain rate of each attribute in each time they need to split, and then to select the attribute which gain rate is maximum to split.

3) C4.5 algorithm:

There is a problem which is biased towards multi-valued attributes in the ID3 algorithm. For example, if there is a unique identity attribute ID, ID3 will choose it as a split attribute, although it makes the division fully pure, but this division is almost useless for classification. ID3's successor algorithm C4.5 uses the information gain expansion of the gain ratio to try to overcome this bias.

C4.5 algorithm first defines the "split information", which definition can be expressed as:

$$\text{split}_{\text{info}}(D) = -\sum_{|D_j|} \frac{|D_j|}{|D|} \log_2 \left( \frac{|D_j|}{|D|} \right)$$  \hspace{1cm} (4)

Among them, the definitions of the symbols are the same as the ID3 algorithm. The gain ratio is defined as:

$$\text{gain\_ratio}(A) = \frac{\text{gain}(A)}{\text{split}_{\text{info}}(A)}$$  \hspace{1cm} (5)

C4.5 also selects the attribute whose gain ratio is maximum as the split attribute. ID3 and C4.5 are more classic decision tree algorithms, which are widely used among people. And the decision tree algorithm has the advantages of high classification precision, small computation, simple generation pattern and so on. It is an excellent classification algorithm.

4) Naive Bayesian algorithm

Naive Bayesian's ideological basis is: for the given items to be classified, calculate the probability of occurrence of each category under this condition, and the maximum probability item is assigned to a specific category. Definition of naive Bayesian classification:

a) Set \(x = \{a_1, a_2, ..., a_m\}\) as a item to be classified, and each \(a\) is a characteristic attribute.

b) A set with category \(C = \{y_1, y_2, ..., y_n\}\)

c) Calculate \(P(y_1|x), P(y_2|x), ..., P(y_n|x)\).

d) If \(P(y_k|x) = \max(P(y_1|x), P(y_2|x), ..., P(y_n|x))\), calculate the probability of each condition in step 3:

- Find a collection of items to be classified, and this set is called the training sample set.
- Obtain the conditional probability estimation of each characteristic attribute under each category by statistics. which is:

\[
P(a_1|y_1), P(a_2|y_1), ..., P(a_m|y_1) : P(a_1|y_2), P(a_2|y_2), ..., P(a_m|y_2) :
\]

...: \(P(a_1|y_n), P(a_2|y_n), ..., P(a_m|y_n)\)  \hspace{1cm} (6)

- If each characteristic attribute is conditional independent, then we will have the following derivation by the Bayesian theorem:

\[
P(y|x) = \frac{P(x|y)P(y)}{P(x)}
\]  \hspace{1cm} (7)

Because the denominator is constant for all categories, we can maximize the numerator. And because each characteristic attribute is conditional independent:

\[
P(x|y)P(y) = P(a_1|y_1)P(a_2|y_2)...P(a_m|y_n)P(y) = P(y) \prod_{i=1}^{m} P(a_i|y_i)
\]  \hspace{1cm} (8)

B. Integrated learning

1) Boosting algorithm

The Boosting algorithm is a method of integrating several classifiers into a classifier. The most famous and most classic one among them is AdaBoost algorithm.

Basic idea of Boosting algorithm is:

a) give the same probability to each training sample first.

b) carry out T iterations. After each iteration, adding weight of the sample of the wrong category (re-sampling) to pay more attention to these samples in the next iteration.

2) AdaBoost algorithm

AdaBoost is an iterative algorithm. And the core is training different classifier for the same training set, which is weak classifier, and then combing the classifiers obtained in each training together, as the final decision classifier.

The algorithm is to change the data distribution. It determines the weight of each sample based on the classification of sample in each training set and the accuracy of the last overall classification. The new data of the modified weights is given to the lower classifier for training, and then the classifier obtained in each training is merged as the final decision classifier.

a) Initialize the weight of all training samples with \(1/N\), \(N\) is the number of samples.

b) for \(m = 1, ..., M\):

- Training the weak classifier \(y_m()\) to minimize the weight error function:

\[
\epsilon_m = \sum_{n=1}^{N} \omega_n^{(m)}(y_m(x_n) \neq t_n)
\]  \hspace{1cm} (9)

- Then calculate the speaking right of this weak classifier \(\alpha\):
\[ \alpha_m = \ln \left\{ \frac{1 - s_m}{s_m} \right\} \]  

- Update weight:
\[ w_{m+1} = \frac{w_m \exp(-\alpha_m t_m y_m(x_i))}{Z_m}, i = 1,2, \ldots, N \]  

Among them, \( Z_m \) is the normalization factor, which let the sum of all \( w \) be 1.
\[ Z_m = \sum_{i=1}^{N} w_m \exp(-\alpha_m t_m y_m(x_i)) \]  

c) get the final classifier:
\[ Y_m(x) = \text{sign} \left( \sum_{m=1}^{M} \alpha_m y_m(x) \right) \]  

IV. INTENT COMPONENT DETECTION MODEL BASED ON MACHINE LEARNING

A. The Principles of Design

All the components and permissions in Android application development are all required to apply in advance, which is set by Android system permissions mechanism. Therefore, for a specific application, we can first decode its apk file, and then perform static analysis of the Androidmanifest.xml to obtain the components it creates and permissions it applies. Then we can extract the effective characteristics of the components and permissions of each application, and form the test samples. After classification prediction algorithm, we first perform a static judgment. Then, the dynamic fuzzy test is carried out on the acquired components, and the different components in the application trigger the corresponding action by randomly generating and sending multiple Intents for different components. Obtaining the system status from logging is to analyze and judge whether there is security vulnerability or not. This model makes full use of the machine learning algorithm. It realizes the combination of static analysis and dynamic analysis, combines two kinds of judgments, and realizes the effective Intent mechanism security vulnerability detection.

B. The design of Model

1) Adaboost algorithm implementation

The implementation of classifier of decision tree. Get the output according to the comparison of the input value and the threshold. The key code is shown in Fig.4.

2) Static analysis

First of all, first perform static analysis of the application. the main task is to extract the information of the components in the applications and the application permissions, and to make the extracted information of the components as the test data input into the integration algorithm for classification testing.

The installation package of Android software follows a certain format. If it is decompressed, we can get a certain directory structure, which has an Androidmanifest.xml. The xml records all the configuration information of this application, including names of the registered components, the information of Intent -filter, the information of permission and so on. For applications that have already been installed, we can analyze the information of the installed applications by reading the package.xml in the / data / system directory in the Android
system. And then we perform static analysis based on the classifier of decision tree in machine learning for the extracted information to detect whether there are component or permission bypass vulnerabilities or deny service vulnerabilities or not, and record the information.

Figure 8. The example of Androidmanifest.xml file

3) Dynamic Analysis

The second step is to perform dynamic analysis of the application, and the main task is to perform dynamic fuzzy test of the components and permissions of the application. The different components of the application trigger the corresponding action by randomly generating and sending multiple Intents for different components. And obtaining the system status from logging is to analyze and judge whether to have implemented permission bypass vulnerabilities or denial of service vulnerabilities or not.

According to the information of component and the information of Intent-filter that is extracted from the static analysis, this module will randomly generate the corresponding Intent to send and to perform the fuzzy test. And then this module will collect the logging information of the system in real time to judge whether to have implemented permission bypass vulnerabilities or denial of service vulnerabilities or not.

The result recording of dynamic analysis comparing with the judgment based on static analysis of machine learning is to generate the final test results.

C. Model test

The experimental applications all run successfully in the Android 4.0 system simulator. Benign samples are all taken from the major domestic application market and passed the major security software detection. Malicious samples are taken from the results of the major security software detection. We collected 150 applications with Android Intent mechanism security vulnerabilities, and 150 applications without Android Intent mechanism security vulnerabilities.

Normal applications and applications with vulnerabilities are randomly divided into three groups. Each group has 50 normal applications and 50 applications with vulnerabilities. First, we randomly selected a group as a sample training, and then chosen the remaining four groups without participating in the training for sample testing. There are three rounds of experiments. The results are shown in Table II.

The evaluation indicators are as follows:

TP - true positive, is the number of vulnerability samples detected correctly
FN - false negative, is the number of vulnerability samples detected falsely
TPR - true positive rate (TPR), the formula is $TPR = TP / (TP + FN)$.
FP - false positive, is the number of normal samples detected falsely
TN - true negative, is the number of normal samples detected correctly
FPR - false positive rate (1-specificity = FPR), the formula is $FPR = FP / (FP + TN)$.
ACC - accuracy

<table>
<thead>
<tr>
<th>Number of rounds</th>
<th>TP</th>
<th>FN</th>
<th>TN</th>
<th>FP</th>
<th>TPR</th>
<th>FPR</th>
<th>ACC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training data</td>
<td>1</td>
<td>38</td>
<td>12</td>
<td>35</td>
<td>14</td>
<td>76%</td>
<td>28%</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>33</td>
<td>17</td>
<td>40</td>
<td>10</td>
<td>66%</td>
<td>20%</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>43</td>
<td>7</td>
<td>41</td>
<td>9</td>
<td>86%</td>
<td>18%</td>
</tr>
<tr>
<td>New sample test</td>
<td>1</td>
<td>34</td>
<td>16</td>
<td>32</td>
<td>18</td>
<td>68%</td>
<td>36%</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>37</td>
<td>13</td>
<td>36</td>
<td>14</td>
<td>74%</td>
<td>28%</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>39</td>
<td>11</td>
<td>38</td>
<td>12</td>
<td>78%</td>
<td>24%</td>
</tr>
</tbody>
</table>

D. Result analysis

Table II shows the results of the training data and the new sample test. According to the results above, this system is good for the training samples classification, with an average accuracy of up to 77 percent. The accuracy of the new sample is not stable, because the number of samples is not large enough. However, compared with the traditional detection system, this experimental result showed that the machine-based Intent mechanism security detection system has certain advantages in accuracy and efficiency. The traditional detection system has a lower recognition rate, even less than 50 percent. If based on a larger sample of data, I believe that our experimental model could achieve higher accuracy of malware identification, and this model will be better optimized.

V. SUMMARY AND OUTLOOK

With the rapid development of mobile Internet, the rapid spread of mobile terminals and the constant enhancement of Internet services and applications, intelligent terminal have become an irreplaceable auxiliary tool. But they also bring many troubles and losses to users because of security vulnerabilities.

Therefore, Android Intent mechanism is widely used and very important. In recent years, the number of denial of service
vulnerabilities or permissions bypass vulnerabilities caused by Intent mechanism has a continued growth. At present, the security research for Android Intent mechanism is mainly based on single dynamic analysis or static analysis. The realization of program is more complex or ineffective.

Based on the above shortcomings, the following studies are carried out in this paper:
1) Analyze and study the Android system and its mechanism.
2) Explore the security problem of the Intent mechanism.
3) Analyze and study the classification algorithm in machine learning.
4) Based on the classification algorithm of machine learning proposed a safety detection model that combines static analysis and dynamic analysis.

This model is based on the combination of static detection and dynamic detection, and it is supplemented by machine learning detector for classification detection. This model could be a better implementation of security vulnerability detection to help developers and Android project team to carry out security testing and maintenance better.

The security detection model of Android Intent mechanism based on machine learning that this paper proposed has a certain degree of innovation and practicability, and it has achieved some achievements. But there are still many shortcomings, which need further study and improvement.

1) The sample data of machine learning algorithm is not enough. We need to continue to do the work in terms of quantity and classification in order to achieve better performance.
2) Fuzzy test module in dynamic detection can join the machine learning algorithm, so that it can generate the Intent which is closer to the actual situation.

REFERENCES